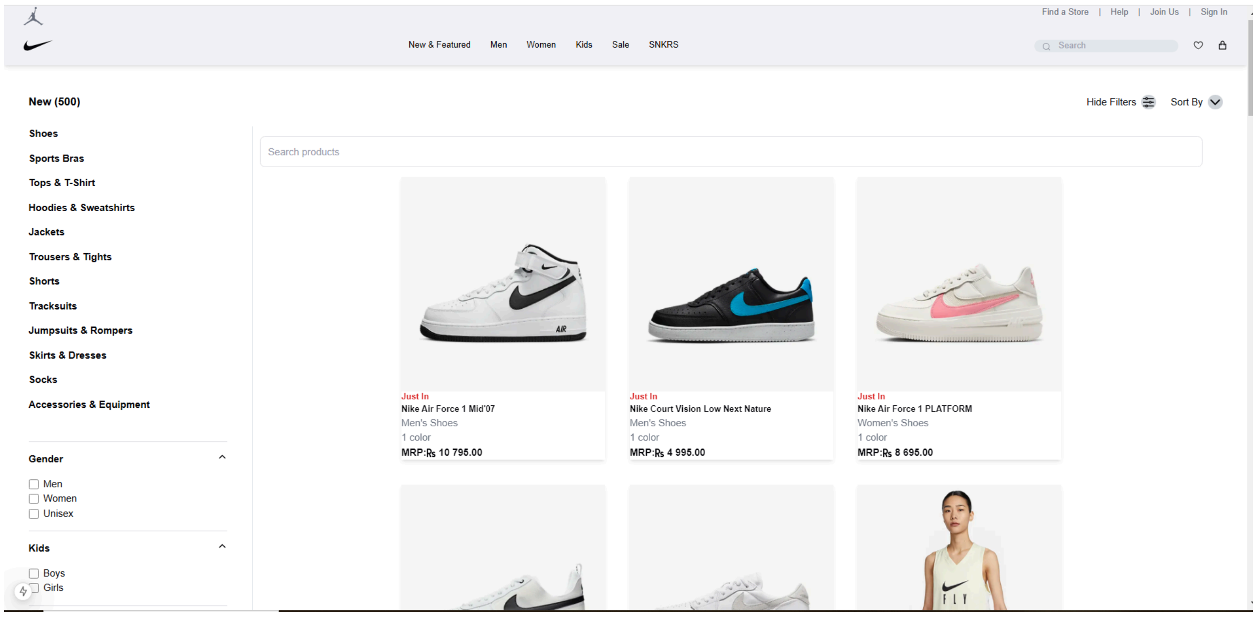
DAY 4: HACKATHON 3

" Building Dynamic Frontend Components for Your Marketplace"

Key Components:

1. Product Listing:

Snippet:



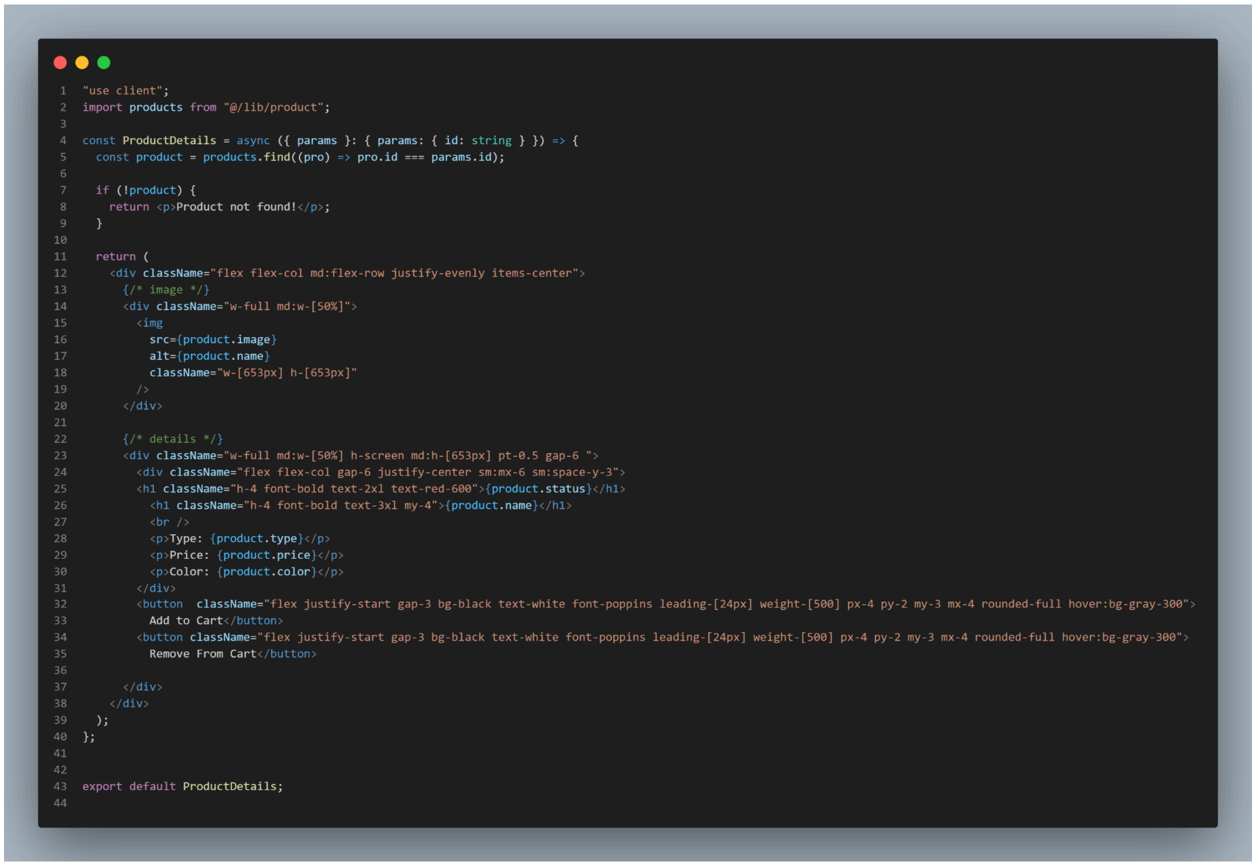
Just In
Nike Air Force 1 Mid'07
Men's Shoes
1 color
MRP:₨ 10 795.00

Just In
Nike Court Vision Low Next Nature
Men's Shoes
1 color
MRP:₨ 4 995.00

Just In
Nike Air Force 1 PLATFORM
Women's Shoes
1 color
MRP:₨ 8 695.00

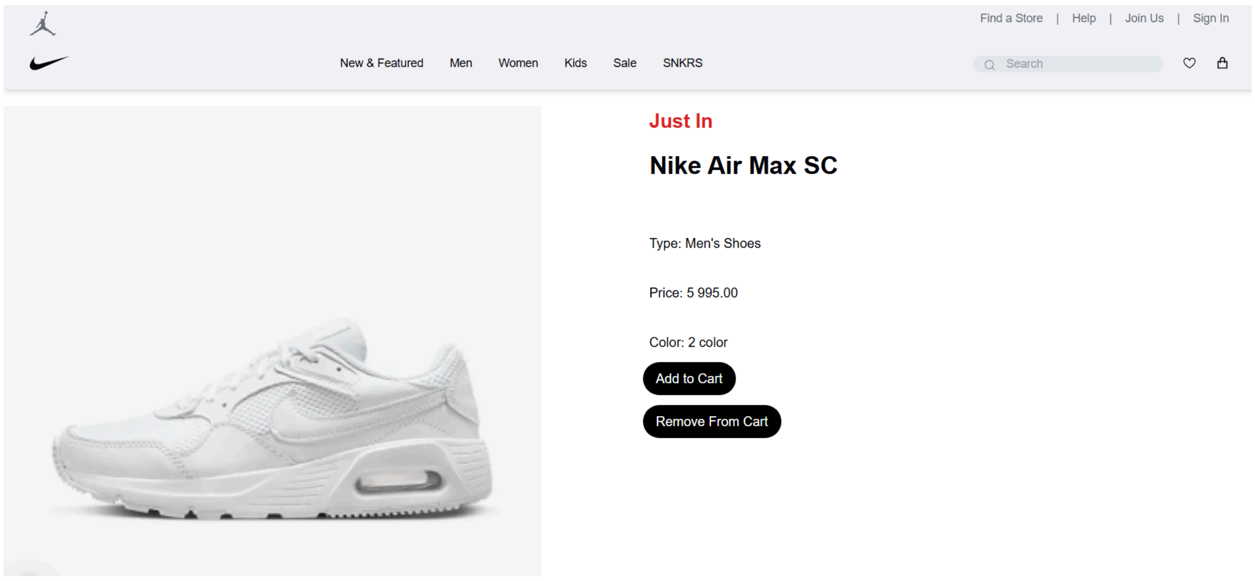
2. Product Detail (dynamic route):

Code snippet:



```tsx
"use client";
import products from "@/lib/product";

const ProductDetails = async ({ params }: { params: { id: string } }) => {
  const product = products.find((pro) => pro.id === params.id);

  if (!product) {
    return <p>Product not found!</p>;
  }

  return (
    <div className="flex flex-col md:flex-row justify-evenly items-center">
      {/* image */}
      <div className="w-full md:w-[50%]">
        <img
          src={product.image}
          alt={product.name}
          className="w-[653px] h-[653px]"
        />
      </div>

      {/* details */}
      <div className="w-full md:w-[50%] h-screen md:h-[653px] pt-0.5 gap-6 ">
        <div className="flex flex-col gap-6 justify-center sm:mx-6 sm:space-y-3">
        <h1 className="h-4 font-bold text-2xl text-red-600">{product.status}</h1>
          <h1 className="h-4 font-bold text-3xl my-4">{product.name}</h1>
          <br />
          <p>Type: {product.type}</p>
          <p>Price: {product.price}</p>
          <p>Color: {product.color}</p>
        </div>
        <button  className="flex justify-start gap-3 bg-black text-white font-poppins leading-[24px] weight-[500] px-4 py-2 my-3 mx-4 rounded-full hover:bg-gray-300">
          Add to Cart</button>
        <button className="flex justify-start gap-3 bg-black text-white font-poppins leading-[24px] weight-[500] px-4 py-2 my-3 mx-4 rounded-full hover:bg-gray-300">
          Remove From Cart</button>

      </div>
    </div>
  );
};


export default ProductDetails;
```

Snippet:



Just In

Nike Air Max SC

Type: Men's Shoes

Price: 5 995.00

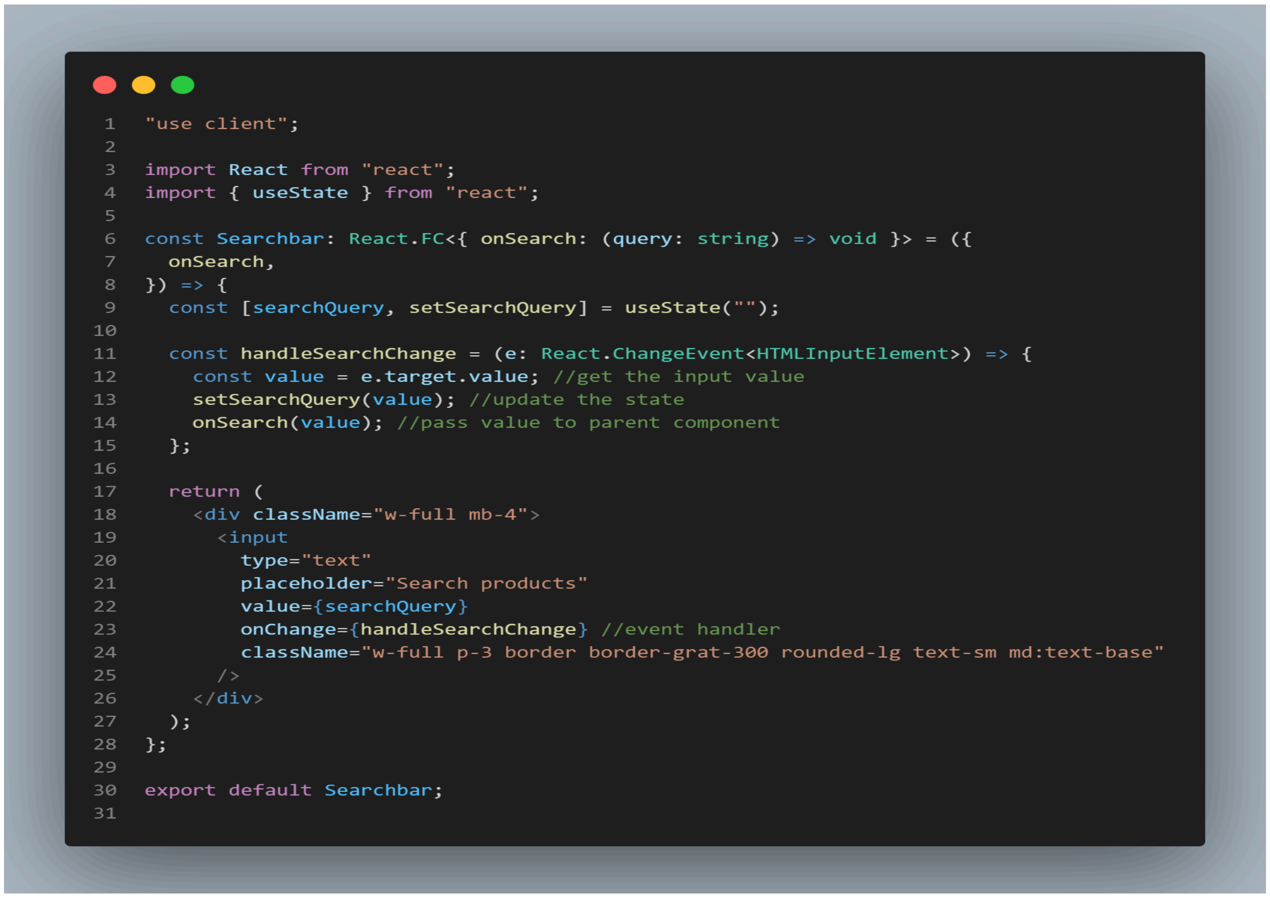
Color: 2 color

Add to Cart

Remove From Cart

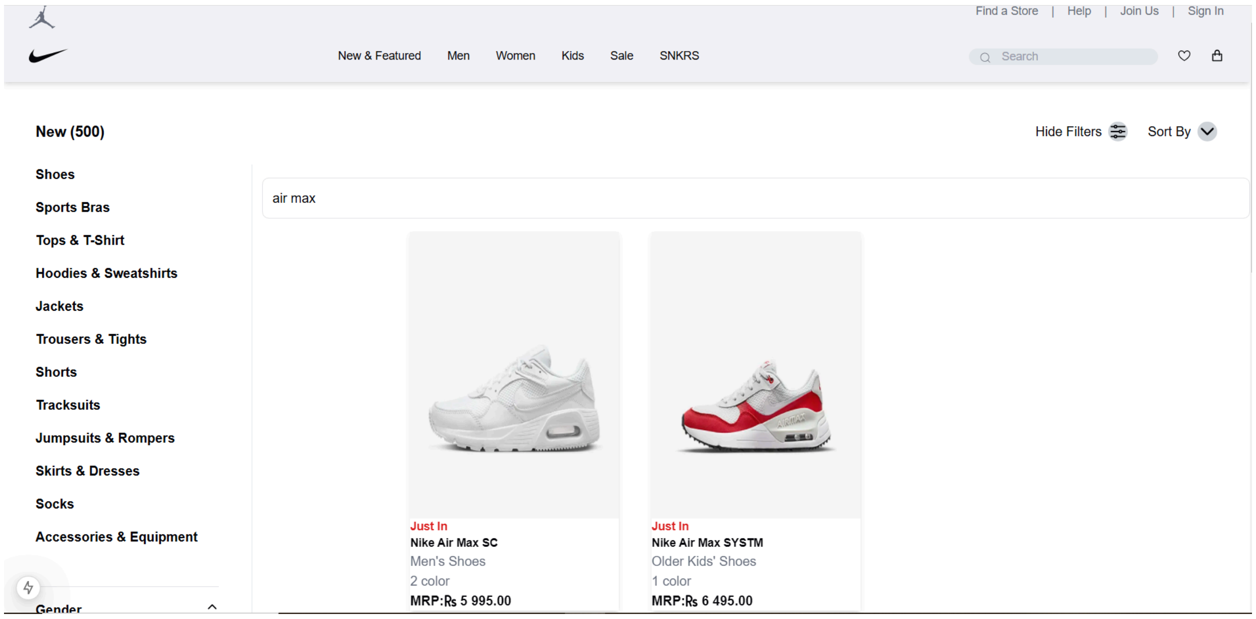
3. Searchbar Component:

Code snippet:



```tsx
"use client";

import React from "react";
import { useState } from "react";

const Searchbar: React.FC<{ onSearch: (query: string) => void }> = ({
  onSearch,
}) => {
  const [searchQuery, setSearchQuery] = useState("");

  const handleSearchChange = (e: React.ChangeEvent<HTMLInputElement>) => {
    const value = e.target.value; //get the input value
    setSearchQuery(value); //update the state
    onSearch(value); //pass value to parent component
  };

  return (
    <div className="w-full mb-4">
      <input
        type="text"
        placeholder="Search products"
        value={searchQuery}
        onChange={handleSearchChange} //event handler
        className="w-full p-3 border border-grat-300 rounded-lg text-sm md:text-base"
      />
    </div>
  );
};

export default Searchbar;

```

Snippet:



New (500)

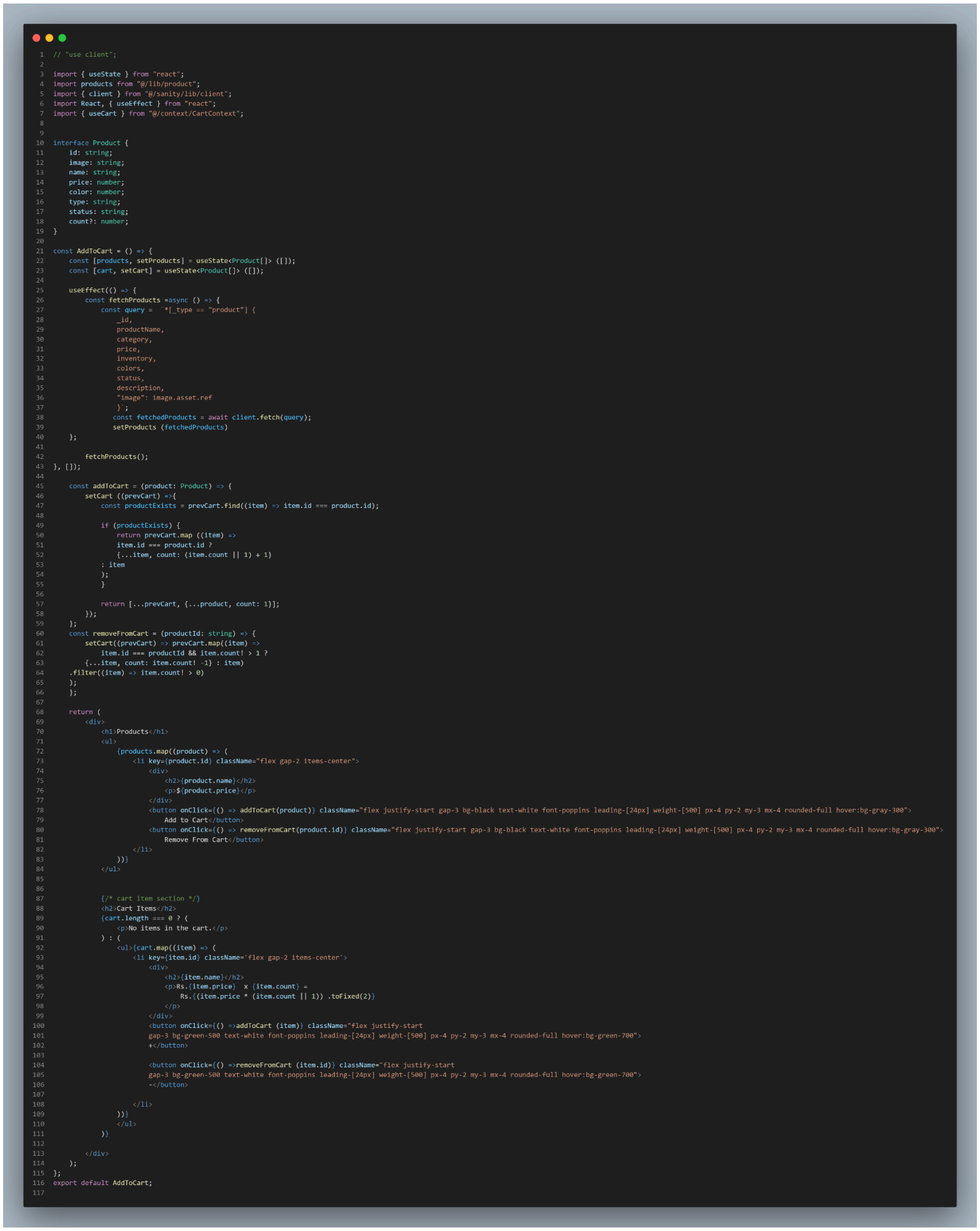
air max

Just In
Nike Air Max SC
Men's Shoes
2 color
MRP:₨ 5 995.00

Just In
Nike Air Max SYSTM
Older Kids' Shoes
1 color
MRP:₨ 6 495.00

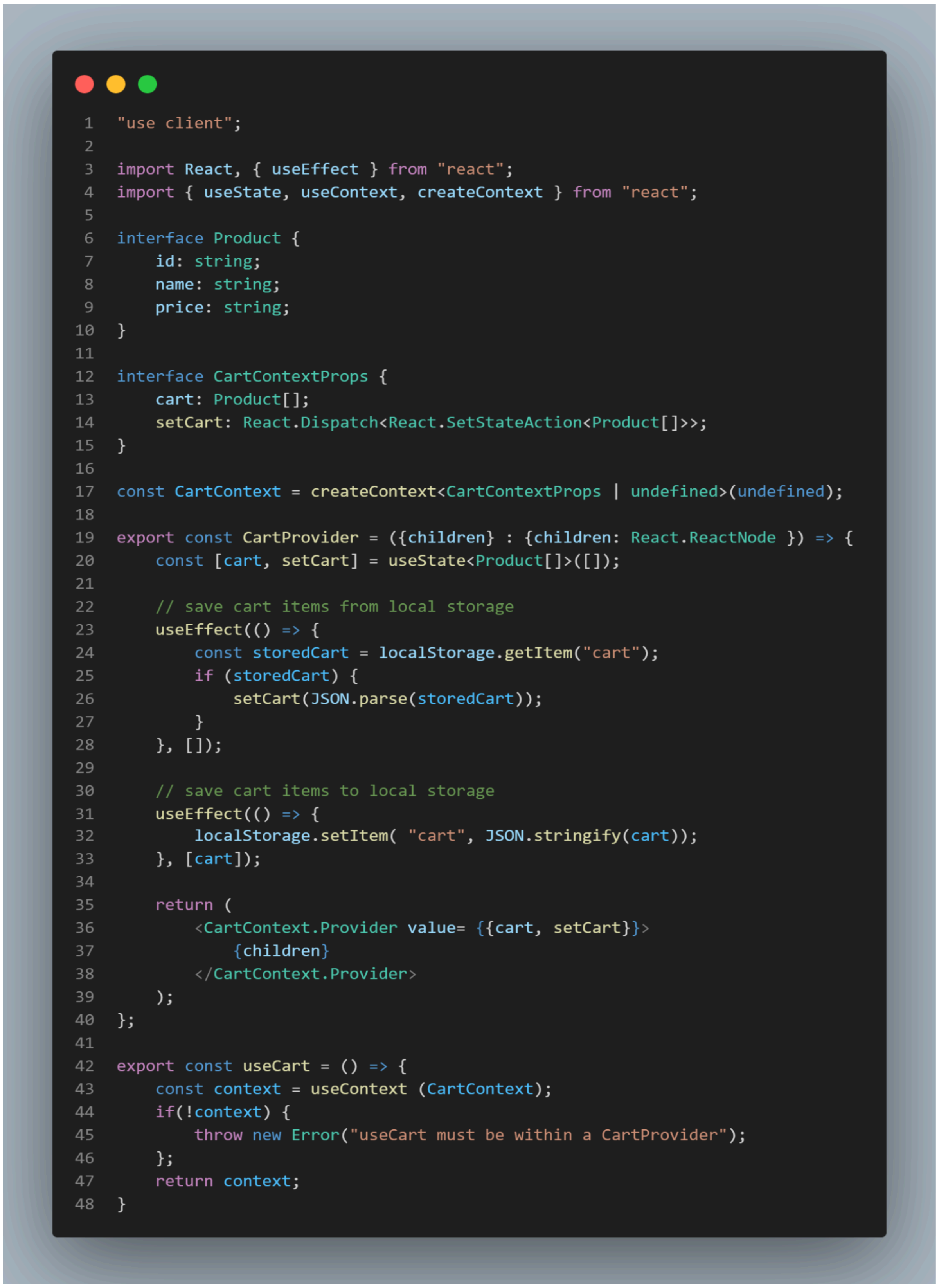
4. Add to Cart Component:

Code snippet (AddToCart):



```tsx
// "use client";

import { useState } from "react";
import products from "@/lib/product";
import { client } from "@/sanity/lib/client";
import React, { useEffect } from "react";
import { useCart } from "@/context/CartContext";


interface Product {
    id: string;
    image: string;
    name: string;
    price: number;
    color: number;
    type: string;
    status: string;
    count?: number;
}

const AddToCart = () => {
    const [products, setProducts] = useState<Product[]> ([]);
    const [cart, setCart] = useState<Product[]> ([]);

    useEffect(() => {
        const fetchProducts =async () => {
            const query =  `*[_type == "product"] {
                _id,
                productName,
                category,
                price,
                inventory,
                colors,
                status,
                description,
                "image": image.asset.ref
                }`;
               const fetchedProducts = await client.fetch(query);
               setProducts (fetchedProducts)
    };

        fetchProducts();
}, []);

    const addToCart = (product: Product) => {
        setCart ((prevCart) =>{
            const productExists = prevCart.find((item) => item.id === product.id);

            if (productExists) {
                return prevCart.map ((item) =>
                item.id === product.id ?
                {...item, count: (item.count || 1) + 1}
            : item
            );
            }

            return [...prevCart, {...product, count: 1}];
        });
    };
    const removeFromCart = (productId: string) => {
        setCart((prevCart) => prevCart.map((item) =>
            item.id === productId && item.count! > 1 ?
        {...item, count: item.count! -1} : item)
    .filter((item) => item.count! > 0)
    );
    };

    return (
        <div>
            <h1>Products</h1>
            <ul>
                {products.map((product) => (
                    <li key={product.id} className="flex gap-2 items-center">
                        <div>
                            <h2>{product.name}</h2>
                            <p>${product.price}</p>
                        </div>
                        <button onClick={() => addToCart(product)} className="flex justify-start gap-3 bg-black text-white font-poppins leading-[24px] weight-[500] px-4 py-2 my-3 mx-4 rounded-full hover:bg-gray-300">
                            Add to Cart</button>
                        <button onClick={() => removeFromCart(product.id)} className="flex justify-start gap-3 bg-black text-white font-poppins leading-[24px] weight-[500] px-4 py-2 my-3 mx-4 rounded-full hover:bg-gray-300">
                            Remove From Cart</button>
                    </li>
                ))}
            </ul>


            {/* cart item section */}
            <h2>Cart Items</h2>
            {cart.length === 0 ? (
                <p>No items in the cart.</p>
            ) : (
                <ul>{cart.map((item) => (
                    <li key={item.id} className='flex gap-2 items-center'>
                        <div>
                            <h2>{item.name}</h2>
                            <p>Rs.{item.price}  x {item.count} =
                                Rs.{(item.price * (item.count || 1)) .toFixed(2)}
                            </p>
                        </div>
                        <button onClick={() =>addToCart (item)} className="flex justify-start
                        gap-3 bg-green-500 text-white font-poppins leading-[24px] weight-[500] px-4 py-2 my-3 mx-4 rounded-full hover:bg-green-700">
                        +</button>

                        <button onClick={() =>removeFromCart (item.id)} className="flex justify-start
                        gap-3 bg-green-500 text-white font-poppins leading-[24px] weight-[500] px-4 py-2 my-3 mx-4 rounded-full hover:bg-green-700">
                        -</button>

                    </li>
                ))}
                </ul>
            )}

        </div>
    );
};
export default AddToCart;

```

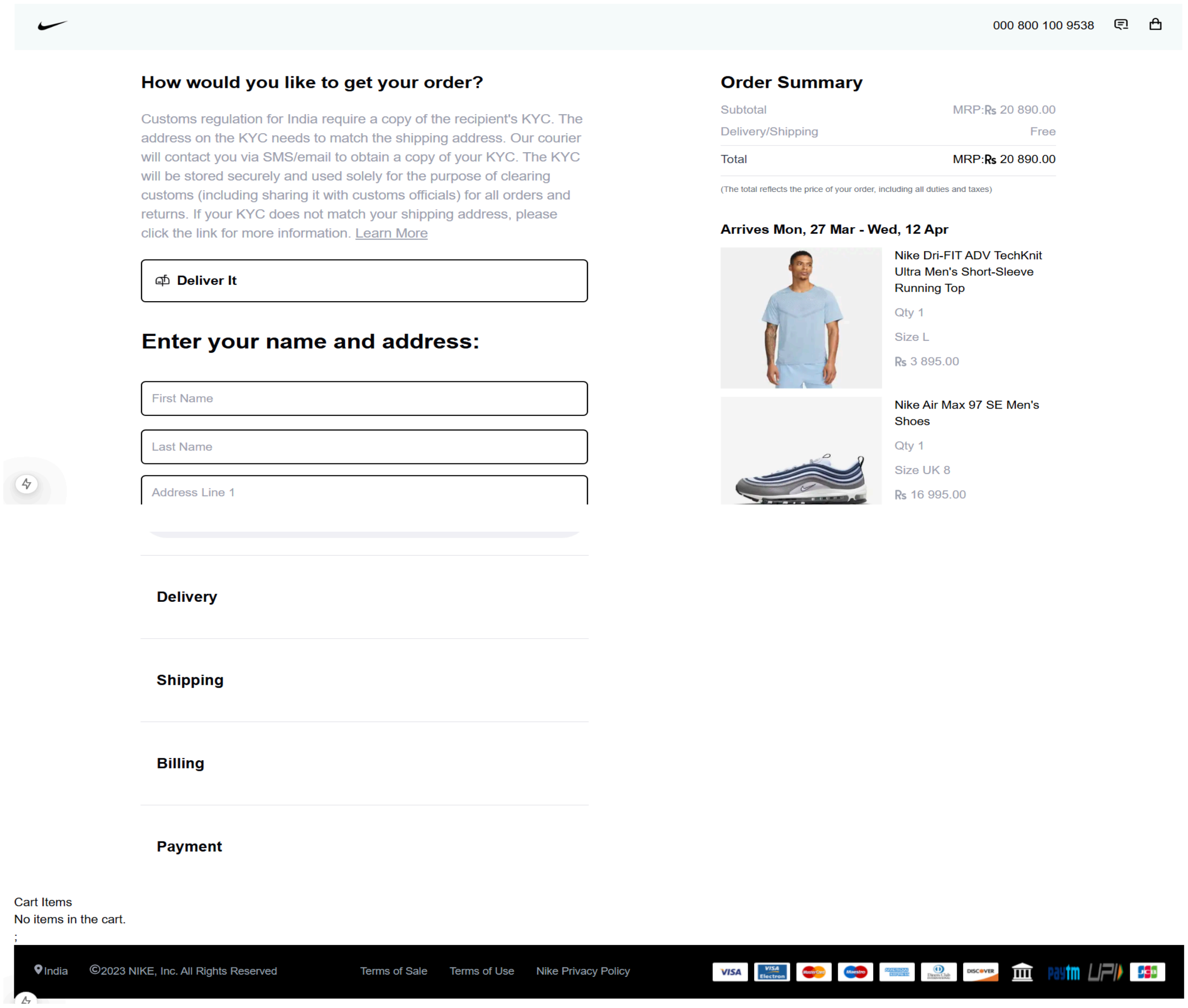
Code snippet (CartContext):



```tsx
"use client";

import React, { useEffect } from "react";
import { useState, useContext, createContext } from "react";

interface Product {
    id: string;
    name: string;
    price: string;
}

interface CartContextProps {
    cart: Product[];
    setCart: React.Dispatch<React.SetStateAction<Product[]>>;
}

const CartContext = createContext<CartContextProps | undefined>(undefined);

export const CartProvider = ({children} : {children: React.ReactNode }) => {
    const [cart, setCart] = useState<Product[]>([]);

    // save cart items from local storage
    useEffect(() => {
        const storedCart = localStorage.getItem("cart");
        if (storedCart) {
            setCart(JSON.parse(storedCart));
        }
    }, []);

    // save cart items to local storage
    useEffect(() => {
        localStorage.setItem( "cart", JSON.stringify(cart));
    }, [cart]);

    return (
        <CartContext.Provider value= {{cart, setCart}}>
            {children}
        </CartContext.Provider>
    );
};

export const useCart = () => {
    const context = useContext (CartContext);
    if(!context) {
        throw new Error("useCart must be within a CartProvider");
    };
    return context;
}
```

5. Checkout Component:

Snippet:



000 800 100 9538

How would you like to get your order?

Customs regulation for India require a copy of the recipient's KYC. The address on the KYC needs to match the shipping address. Our courier will contact you via SMS/email to obtain a copy of your KYC. The KYC will be stored securely and used solely for the purpose of clearing customs (including sharing it with customs officials) for all orders and returns. If your KYC does not match your shipping address, please click the link for more information. Learn More

Enter your name and address:

Order Summary

Subtotal	MRP:₨ 20 890.00
Delivery/Shipping	Free
Total	MRP:₨ 20 890.00

(The total reflects the price of your order, including all duties and taxes)

Arrives Mon, 27 Mar - Wed, 12 Apr

Nike Dri-FIT ADV TechKnit Ultra Men's Short-Sleeve Running Top
Qty 1
Size L
₨ 3 895.00

Nike Air Max 97 SE Men's Shoes
Qty 1
Size UK 8
₨ 16 995.00

Cart Items
No items in the cart.
;

India ©2023 NIKE, Inc. All Rights Reserved Terms of Sale Terms of Use Nike Privacy Policy

Final Checklist:

Frontend Component Development:	Styling and Responsiveness:	Code Quality:	Documentation and Submission:	Final Review:
✔	✔	✔	✔	✔